Framingham Dataset

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [ ]:
```
df=pd.read_csv('data/framingham.csv')
df.head(20)
```

Out[ ]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP
0	1	39	4.0	0	0.0	0.0	0	0	0	195.0	106.0	70.0
1	0	46	2.0	0	0.0	0.0	0	0	0	250.0	121.0	81.0
2	1	48	1.0	1	20.0	0.0	0	0	0	245.0	127.5	80.0
3	0	61	3.0	1	30.0	0.0	0	1	0	225.0	150.0	95.0
4	0	46	3.0	1	23.0	0.0	0	0	0	285.0	130.0	84.0
5	0	43	2.0	0	0.0	0.0	0	1	0	228.0	180.0	110.0
6	0	63	1.0	0	0.0	0.0	0	0	0	205.0	138.0	71.0
7	0	45	2.0	1	20.0	0.0	0	0	0	313.0	100.0	71.0
8	1	52	1.0	0	0.0	0.0	0	1	0	260.0	141.5	89.0
9	1	43	1.0	1	30.0	0.0	0	1	0	225.0	162.0	107.0
10	0	50	1.0	0	0.0	0.0	0	0	0	254.0	133.0	76.0
11	0	43	2.0	0	0.0	0.0	0	0	0	247.0	131.0	88.0
12	1	46	1.0	1	15.0	0.0	0	1	0	294.0	142.0	94.0
13	0	41	3.0	0	0.0	1.0	0	1	0	332.0	124.0	88.0
14	0	39	2.0	1	9.0	0.0	0	0	0	226.0	114.0	64.0
15	0	38	2.0	1	20.0	0.0	0	1	0	221.0	140.0	90.0
16	1	48	3.0	1	10.0	0.0	0	1	0	232.0	138.0	90.0
17	0	46	2.0	1	20.0	0.0	0	0	0	291.0	112.0	78.0
18	0	38	2.0	1	5.0	0.0	0	0	0	195.0	122.0	84.5
19	1	41	2.0	0	0.0	0.0	0	0	0	195.0	139.0	88.0

In [ ]:
```
df.nunique()
```

Out[ ]:
```
male                2
age                39
education           4
currentSmoker       2
cigsPerDay         33
BPMeds              2
prevalentStroke     2
prevalentHyp        2
diabetes            2
totChol           248
sysBP             234
diaBP             146
BMI              1363
heartRate          73
glucose           143
TenYearCHD          2
dtype: int64
```

In [ ]:
```
df.isnull().sum()
```

Out[ ]:
```
male                0
age                 0
education         105
currentSmoker       0
cigsPerDay         29
BPMeds             53
prevalentStroke     0
prevalentHyp        0
diabetes            0
totChol            50
sysBP               0
diaBP               0
BMI                19
heartRate           1
glucose           388
TenYearCHD          0
dtype: int64
```

In [ ]:
```
df.duplicated().any()
```

Out[ ]:
```
False
```

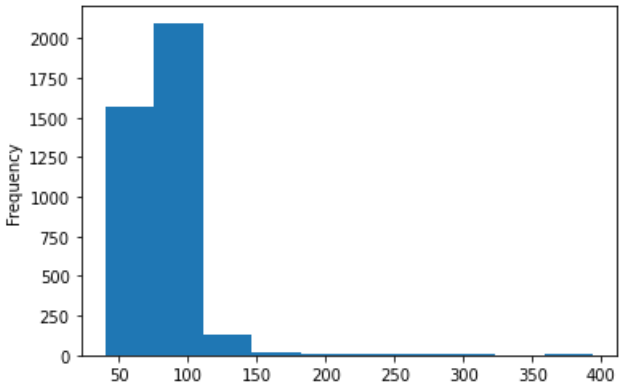
In [ ]:
```
df.describe()
```

Out[ ]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabet
count	4238.000000	4238.000000	4133.000000	4238.000000	4209.000000	4185.000000	4238.000000	4238.000000	4238.0000
mean	0.429212	49.584946	1.978950	0.494101	9.003089	0.029630	0.005899	0.310524	0.0257
std	0.495022	8.572160	1.019791	0.500024	11.920094	0.169584	0.076587	0.462763	0.1583
min	0.000000	32.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
25%	0.000000	42.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
50%	0.000000	49.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
75%	1.000000	56.000000	3.000000	1.000000	20.000000	0.000000	0.000000	1.000000	0.0000
max	1.000000	70.000000	4.000000	1.000000	70.000000	1.000000	1.000000	1.000000	1.0000

In [ ]:
```
df['glucose'].plot.hist()
```

Out[ ]:
```
<AxesSubplot: ylabel='Frequency'>
```



labels = 0.male 1.age 2.education 3.currentSmoker 4.cigsPerDay 5.BPMeds 6.prevalentStroke 7.prevalentHyp 8.diabetes 9.totChol 10.sysBP 11.diaBP 12.BMI 13.heartRate 14.glucose 15.TenYearCHD

Mean - totChol, BMI (columns:9,12)

Mode - Education, cigsPerDay, BPMeds (columns:2,4,5)

Median - glucose (columns:14)

Scaler - columns:1,2,4,9,10,11,12,13,14

In [ ]:
```
df.dropna(inplace=True, subset='heartRate')
```

In [ ]:
```
df.isnull().sum()
```

Out[ ]:
```
male                0
age                 0
education         105
currentSmoker       0
cigsPerDay         29
BPMeds             53
prevalentStroke     0
prevalentHyp        0
diabetes            0
totChol            50
sysBP               0
diaBP               0
BMI                19
heartRate           0
glucose           388
TenYearCHD          0
dtype: int64
```

In [ ]:
```
data = df.values
X = data[:, :-1]
y = data[:, -1]
pd.DataFrame(X)
```

Out[ ]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	1.0	39.0	4.0	0.0	0.0	0.0	0.0	0.0	0.0	195.0	106.0	70.0	26.97	80.0	77.0
1	0.0	46.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	250.0	121.0	81.0	28.73	95.0	76.0
2	1.0	48.0	1.0	1.0	20.0	0.0	0.0	0.0	0.0	245.0	127.5	80.0	25.34	75.0	70.0
3	0.0	61.0	3.0	1.0	30.0	0.0	0.0	1.0	0.0	225.0	150.0	95.0	28.58	65.0	103.0
4	0.0	46.0	3.0	1.0	23.0	0.0	0.0	0.0	0.0	285.0	130.0	84.0	23.10	85.0	85.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4232	1.0	50.0	1.0	1.0	1.0	0.0	0.0	1.0	0.0	313.0	179.0	92.0	25.97	66.0	86.0
4233	1.0	51.0	3.0	1.0	43.0	0.0	0.0	0.0	0.0	207.0	126.5	80.0	19.71	65.0	68.0
4234	0.0	48.0	2.0	1.0	20.0	NaN	0.0	0.0	0.0	248.0	131.0	72.0	22.00	84.0	86.0
4235	0.0	44.0	1.0	1.0	15.0	0.0	0.0	0.0	0.0	210.0	126.5	87.0	19.16	86.0	NaN
4236	0.0	52.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	269.0	133.5	83.0	21.47	80.0	107.0

4237 rows × 15 columns

In [ ]:
```
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import ConfusionMatrixDisplay,confusion_matrix, accuracy_score,precision_score
```

In [ ]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=1)
```

In [ ]:
```
mean_imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
median_imputer = SimpleImputer(missing_values=np.nan, strategy='median')
mode_imputer = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
min_max_scaler = MinMaxScaler(feature_range=(0,1))
```

labels = 0.male 1.age 2.education 3.currentSmoker 4.cigsPerDay 5.BPMeds 6.prevalentStroke 7.prevalentHyp 8.diabetes 9.totChol 10.sysBP 11.diaBP 12.BMI 13.heartRate 14.glucose 15.TenYearCHD

Mean - totChol, BMI (columns:9,12)

Mode - Education, cigsPerDay, BPMeds (columns:2,4,5)

Median - glucose (columns:14)

Scaler - columns:1,2,4,9,10,11,12,13,14

In [ ]:
```
X_train[:, (9,12)] = mean_imputer.fit_transform(X_train[:, (9,12)])
X_train[:, (14,)] = median_imputer.fit_transform(X_train[:, (14,)])
X_train[:, (2,4,5)] = mode_imputer.fit_transform(X_train[:, (2,4,5)])

X_test[:, (9,12)] = mean_imputer.transform(X_test[:, (9,12)])
X_test[:, (14,)] = median_imputer.transform(X_test[:, (14,)])
X_test[:, (2,4,5)] = mode_imputer.transform(X_test[:, (2,4,5)])
```

In [ ]:
```
pd.DataFrame(X_train).isnull().any()
```

Out[ ]:
```
0     False
1     False
2     False
3     False
4     False
5     False
6     False
7     False
8     False
9     False
10    False
11    False
12    False
13    False
14    False
dtype: bool
```

In [ ]:
```
X_train[:, (1,2,4,9,10,11,12,13,14)] = min_max_scaler.fit_transform(X_train[:, (1,2,4,9,10,11,12,13,14)])
X_test[:, (1,2,4,9,10,11,12,13,14)] = min_max_scaler.transform(X_test[:, (1,2,4,9,10,11,12,13,14)])
```

In [ ]:
```
pd.DataFrame(X_train)
```

Out[ ]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	1.0	0.105263	0.333333	1.0	0.333333	0.0	0.0	0.0	0.0	0.202037	0.249240	0.386243	0.201256	0.242424	0.096045
1	0.0	0.157895	0.333333	1.0	0.166667	0.0	0.0	0.0	0.0	0.342954	0.179331	0.211640	0.192003	0.161616	0.098870
2	0.0	0.500000	0.333333	0.0	0.000000	0.0	0.0	0.0	0.0	0.302207	0.322188	0.407407	0.306345	0.262626	0.121469
3	0.0	0.921053	0.333333	1.0	0.250000	0.0	0.0	1.0	0.0	0.448217	0.501520	0.391534	0.324190	0.565657	0.129944
4	1.0	0.736842	1.000000	1.0	0.050000	0.0	0.0	1.0	0.0	0.271647	0.516717	0.634921	0.367812	0.646465	0.121469
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3384	0.0	0.473684	0.666667	0.0	0.000000	0.0	0.0	0.0	0.0	0.242784	0.155015	0.232804	0.171844	0.313131	0.104520
3385	1.0	0.236842	1.000000	0.0	0.000000	0.0	0.0	0.0	0.0	0.152801	0.306991	0.412698	0.328486	0.313131	0.098870
3386	1.0	0.552632	0.000000	0.0	0.000000	0.0	0.0	0.0	0.0	0.191851	0.264438	0.296296	0.288500	0.313131	0.096045
3387	0.0	0.763158	1.000000	0.0	0.000000	0.0	0.0	1.0	0.0	0.217317	0.750760	0.788360	0.532056	0.363636	0.090395
3388	0.0	0.105263	0.000000	1.0	0.050000	0.0	0.0	0.0	0.0	0.047538	0.148936	0.275132	0.230998	0.292929	0.098870

3389 rows × 15 columns

In [ ]:
```
log_model=LogisticRegression()
log_model.fit(X_train, y_train)
```

Out[ ]:
```
▾ LogisticRegression

LogisticRegression()
```

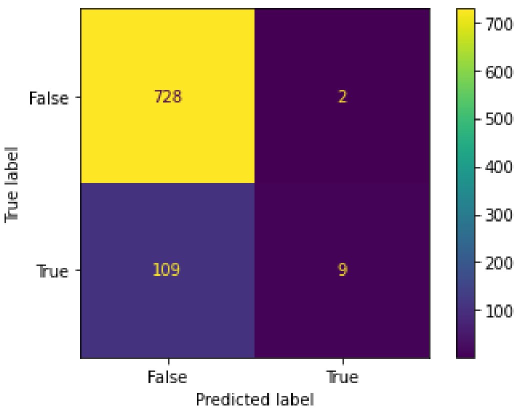
In [ ]:
```
log_pred = log_model.predict(X_test)
log_pred
```

```
Out[ ]: array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

```
In [ ]: confusion_matrix = confusion_matrix(y_test, log_pred)
        confusion_matrix
```

```
Out[ ]: array([[728,   2],
               [109,   9]])
```

```
In [ ]: cm_display = ConfusionMatrixDisplay(confusion_matrix=confusion_matrix, display_labels=[False, True])
        cm_display.plot()
        plt.show()
```




```
In [ ]: accuracy = accuracy_score(y_test, log_pred)
        accuracy
```

```
Out[ ]: 0.8691037735849056
```

In [ ]:
```
precision = precision_score(y_test, log_pred)
precision
```

Out[ ]:
```
0.8181818181818182
```